

Undecidability of model-checking branching-time properties of stateless probabilistic pushdown process

Tianrong Lin*

Abstract

In this paper, we settle a problem in probabilistic verification of infinite-state process (specifically, *probabilistic pushdown process*). We show that model checking *stateless probabilistic pushdown process* (pBPA) against *probabilistic computational tree logic* (PCTL) is undecidable.

Keywords: Probabilistic pushdown process, Undecidability, Probabilistic computational tree logic.

1 Introduction

Model checking, see [3] by Clarke et al., is an essential tool for formal verification, in which one describes the system to be verified as a model of some logic, expresses the property to be verified as a formula in that logic, and then checks by using automated algorithms that the formula holds or not in that model [2] by Baier et al. Traditionally, model checking has been applied to finite-state systems and non-probabilistic programs. To the author’s knowledge, the verification of probabilistic programs was considered first in the 1980s, for example [12] by Vardi. During recent two decades, researchers have paid their attention to model-checking of probabilistic infinite-state systems, for instance [8, 7] by Esparza.

One of such probabilistic infinite-state systems is probabilistic pushdown process, which was called “probabilistic pushdown automata” in [8, 7, 16, 15]. Here, we reserve “probabilistic pushdown automata” for the probabilistic extension of nondeterministic pushdown automata [14, 6]. Roughly, probabilistic pushdown process can be seen as probabilistic pushdown automaton with only a input symbol, which means that it is can be considered as a restricted probabilistic pushdown automaton. Their model-checking problem, initialized by Esparza et al. [8, 7], has attracted a lot of attention, for example [16, 15] by Brázdil et al., in which the model-checking problem of stateless probabilistic pushdown process (pBPA) against PCTL* was resolved, as well as the model-checking of probabilistic pushdown process (pPDS) against PCTL (throughout the paper, for the author’s habit, ‘probabilistic pushdown process’ is just another appellation of ‘probabilistic pushdown automata’ in [16, 15]). On the other hand, the problem of model-checking of stateless probabilistic pushdown process (pBPA) against PCTL remains open in [16, 15], which was first proposed in [7].

This paper aims at providing a solution to that problem. Our main idea here is to further employ the value of the construction presented in [15, 16]. Based on this thought, we attempt to construct PCTL formulas which encode the modified Post Correspondence Problem. We show here that:

Theorem 1.1. *The model-checking of stateless probabilistic pushdown process (pBPA) against probabilistic computational tree logic PCTL is undecidable.*

*E-mail address: tianrong.lam@gmail.com

Because the class of stateless probabilistic pushdown process is a sub-class of probabilistic pushdown process, and the logic of PCTL is a sublogic of PCTL*, by Theorem 1.1 we can re-obtain the undecidability results in [16].

The rest of this paper is structured as follows: in the next Section some basic definitions will be reviewed and useful notations will be fixed. Section 3 is devoted to the proof of the main theorem, and the last Section is for conclusions.

2 Preliminaries

For convenience and purpose of fully exploiting the technique developed in [16, 15], most notations (except some personal preferred) will follow from [16, 15]. In addition, for elementary probability theory, the reader is referred to [1] by Shiryaev, or [10, 11] by Loève.

For any finite set S , $|S|$ denotes the cardinality of S . Throughout this paper, Σ , and Γ denote the non-empty finite alphabets, Σ^* denotes the set of all finite words (including empty word ϵ) over Σ , and $\Sigma^+ = \Sigma^* \setminus \{\epsilon\}$. Let w be a word in Σ^* , then $|w|$ will denote the length of w . For example, let $\Sigma = \{0, 1\}$, then $|\epsilon| = 0$ and $|001101| = 6$.

2.1 Markov Chains

Roughly, Markov chains are probabilistic transition systems which are accepted [2] as the most popular operational model for the evaluation of performance and dependability of information-processing systems.

Definition 2.1. A (discrete) Markov chain is a triple $\mathcal{M} = (S, \delta, \mathcal{P})$ where S is a finite or countably infinite set of states, $\delta \subseteq S \times S$ is a transition relation such that for each $s \in S$ there exists $t \in S$ such that $(s, t) \in \delta$, and \mathcal{P} is a function from domain δ to range $(0, 1]$ which to each transition $(s, t) \in \delta$ assigns its probability $\mathcal{P}(s, t)$ such that $\sum_{(s, t) \in \delta} \mathcal{P}(s, t) = 1$ for all $s \in S$.

A path in \mathcal{M} is a finite or infinite sequence of states of S : $\omega = s_0 s_1 \dots$ such that $(s_i, s_{i+1}) \in \delta$ for each i . A run of \mathcal{M} is an infinite path. We denote the set of all runs in \mathcal{M} by Run , and $Run(\omega')$ to denote the set of all runs starting with a given finite path ω' . Let ω be a given run, then $\omega(i)$ denotes the state s_i of ω , and ω_i the run $s_i s_{i+1} \dots$. In this way, it is clear that $\omega_0 = \omega$. Further, a state s' is *reachable* from a state s if there is a *finite path* starting in s and ending at s' .

For each $s \in S$, $(Run(s), \mathcal{F}, \mathcal{P})$ is a probability space, where \mathcal{F} is the σ -field generated by all *basic cylinders* $Run(\omega)$ where ω is a finite path initiating from s , and $\mathcal{P} : \mathcal{F} \rightarrow [0, 1]$ is the unique probability measure such that $\mathcal{P}(Run(\omega)) = \prod_{1 \leq i \leq |\omega|} \mathcal{P}(s_{i-1}, s_i)$ where $\omega = s_0 s_1 \dots s_{|\omega|}$.

2.2 Probabilistic Computational Tree Logic

The logic PCTL was originally introduced by Hansson et al. in [5], where the corresponding model-checking problem has been focused mainly on finite-state Markov chains.

Let AP be a fixed set of atomic propositions. Formally, the syntax of probabilistic computational tree logic PCTL is defined by

$$\begin{aligned} \Phi &::= p \mid \neg \Phi \mid \Phi_1 \wedge \Phi_2 \mid \mathcal{P}_{\bowtie r}(\varphi) \\ \varphi &::= \mathbf{X}\Phi \mid \Phi_1 \mathbf{U} \Phi_2 \end{aligned}$$

where Φ and φ denote the state formula and path formula respectively; $p \in AP$ is an atomic proposition, $\bowtie \in \{>, =\}$ ¹, r is a rational with $0 \leq r \leq 1$. The symbol **true** is the abbreviation of always true.

¹ We do not include other relations of comparison such as “ \geq ”, “ \leq ”, and “ $<$ ”, because “ $>$ ” and “ $=$ ” are sufficient enough for our discussion.

Let $\mathcal{M} = (S, \delta, \mathcal{P})$ be a Markov chain and $\nu : AP \rightarrow 2^S$ an assignment. Then the semantics of PCTL, over \mathcal{M} , is given by the following rules

$$\begin{aligned}
\mathcal{M}, s \models^\nu \mathbf{true} & \quad \text{for any } s \in S, \\
\mathcal{M}, s \models^\nu p & \quad \Leftrightarrow \quad s \in \nu(p), \\
\mathcal{M}, s \models^\nu \neg\Phi & \quad \Leftrightarrow \quad \mathcal{M}, s \not\models^\nu \Phi, \\
\mathcal{M}, s \models^\nu \Phi_1 \wedge \Phi_2 & \quad \Leftrightarrow \quad \mathcal{M}, s \models^\nu \Phi_1 \text{ and } \mathcal{M}, s \models^\nu \Phi_2, \\
\mathcal{M}, s \models^\nu \mathcal{P}_{\bowtie r}(\varphi) & \quad \Leftrightarrow \quad \mathcal{P}(\{\omega \in \text{Run}(s) : \mathcal{M}, s \models^\nu \varphi\}) \bowtie r, \\
\\
\mathcal{M}, \omega \models^\nu \mathbf{X}\Phi & \quad \Leftrightarrow \quad \mathcal{M}, \omega(1) \models^\nu \Phi, \\
\mathcal{M}, \omega \models^\nu \Phi_1 \mathbf{U} \Phi_2 & \quad \Leftrightarrow \quad \text{for some } k \geq 0 \text{ such that } \mathcal{M}, \omega_k \models^\nu \Phi_2 \text{ and for all } j, \\
& \quad 0 \leq j < k : \mathcal{M}, \omega_j \models^\nu \Phi_1.
\end{aligned}$$

Remark 1. The another probabilistic computational tree logic $PCTL^*$, whose path formula are generated by the following syntax, contains the logic PCTL as a sublogic

$$\varphi ::= \Phi \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \mathbf{X}\varphi \mid \varphi_1 \mathbf{U} \varphi_2.$$

The difference of formulas between PCTL and $PCTL^*$ is very clear: a well-defined formula of PCTL is definitely a well-defined $PCTL^*$ formula, however, the inverse is not necessarily true. The semantics of $PCTL^*$ path formulas are defined, over \mathcal{M} , as follows

$$\begin{aligned}
\mathcal{M}, \omega \models^\nu \Phi & \quad \Leftrightarrow \quad \mathcal{M}, \omega(0) \models^\nu \Phi, \\
\mathcal{M}, \omega \models^\nu \neg\varphi, & \quad \Leftrightarrow \quad \mathcal{M}, \omega \not\models^\nu \varphi \\
\mathcal{M}, \omega \models^\nu \varphi_1 \wedge \varphi_2 & \quad \Leftrightarrow \quad \mathcal{M}, \omega \models^\nu \varphi_1 \text{ and } \mathcal{M}, \omega \models^\nu \varphi_2, \\
\mathcal{M}, \omega \models^\nu \mathbf{X}\varphi & \quad \Leftrightarrow \quad \mathcal{M}, \omega_1 \models^\nu \varphi \\
\mathcal{M}, \omega \models^\nu \varphi_1 \mathbf{U} \varphi_2 & \quad \Leftrightarrow \quad \text{for some } k \geq 0 \text{ s.t. } \mathcal{M}, \omega_k \models^\nu \varphi_2 \text{ and for all } 0 \leq j < k \\
& \quad \text{s.t. } \mathcal{M}, \omega_j \models^\nu \varphi_1.
\end{aligned}$$

Remark 2. The logic of PCTL or $PCTL^*$ can be interpreted over an MDP \mathcal{M} in a similar way we have done in the case of Markov chain.

2.3 Probabilistic pushdown process

Let us recall the definitions of probabilistic pushdown process, being as follows.

Definition 2.2. A probabilistic pushdown process (pPDS) is a tuple $\Delta = (Q, \Gamma, \delta, \mathcal{P})$ where Q is a finite set of control states, Γ a finite stack alphabet, $\delta \subseteq (Q \times \Gamma) \times (Q \times \Gamma^*)$ a finite set of rules satisfying

- for every $(p, X) \in Q \times \Gamma$ there is at least one rule of the form $((p, X), (q, \alpha)) \in \delta$; In the following we will write $(p, X) \rightarrow (q, \alpha)$ instead of $((p, X), (q, \alpha)) \in \delta$.
- \mathcal{P} is a function from δ to $(0, 1]$ which to every rule $(p, X) \rightarrow (q, \alpha)$ in δ assigns its probability $\mathcal{P}((p, X) \rightarrow (q, \alpha)) \in (0, 1]$ s.t. for all $(p, X) \in Q \times \Gamma$ satisfying the following

$$\sum_{(p, X) \rightarrow (q, \alpha)}^{(q, \alpha) \in Q \times \Gamma^*} \mathcal{P}((p, X) \rightarrow (q, \alpha)) = 1$$

Further, without loss of generality, we assume $|\alpha| \leq 2$. The configurations of Δ are elements in $Q \times \Gamma^*$.

The *stateless probabilistic pushdown process* (pPBA) is a *probabilistic pushdown process* (pPDS) whose state set Q is a singleton (or, we even can omit Q without any influence).

Definition 2.3. A *stateless probabilistic pushdown process* (pBPA) is a triple $\Delta = (\Gamma, \delta, \mathcal{P})$, whose configurations are elements $\in \Gamma^*$, where Γ is a finite stack alphabet, δ a finite set of rules satisfies

- for each $X \in \Gamma$ there is at least one rule $(X, \alpha) \in \delta$ where $\alpha \in \Gamma^*$. In the following, we write $X \rightarrow \alpha$ instead of $(X, \alpha) \in \delta$; We assume, w.l.o.g., that $|\alpha| \leq 2$.
- \mathcal{P} is a function from δ to $(0, 1]$ which to every rule $X \rightarrow \alpha$ in δ assigns its probability $\mathcal{P}(X \rightarrow \alpha) \in (0, 1]$ s.t. for all $X \in \Gamma$, it meets

$$\sum_{\substack{\alpha \in \Gamma^* \\ X \rightarrow \alpha}} \mathcal{P}(X \rightarrow \alpha) = 1.$$

Given a pPDS or pBPA Δ , it is not hard to see that all of its configurations with all its transition rules and corresponding probabilities induce an infinite-state Markov chain \mathcal{M}_Δ . The model-checking problem for properties expressed by PCTL formula is defined to decide whether $\mathcal{M}_\Delta \models^\nu \Psi$.

As observed in [9], one can easily encode undecidable properties to pushdown configurations if there is no ‘effective assumptions’ about valuations. Thus we consider the same assignment as [9, 8, 7, 16, 15, 17], which was called ‘regular assignment’. More precisely, let $\Delta = (Q, \Gamma, \delta, \mathcal{P})$ be a probabilistic pushdown process, an assignment $\nu : AP \rightarrow 2^{Q \times \Gamma^*}$ (2^{Γ^*} for pPBA) is regular if $\nu(p)$ is a regular set for each $p \in AP$. In other words, $\nu(p)$ can be recognized by finite automata \mathcal{A}_p over the alphabet $Q \cup \Gamma$, and \mathcal{A}_p reads the stack of Δ from bottom up. Further, the regular assignment ν is simple if for each $p \in AP$ there is a subset of heads $H_p \subseteq Q \cup (Q \times \Gamma)$ s.t. $(q, \gamma\alpha) \in \nu(p) \Leftrightarrow (q, \gamma) \in H_p$ [16].

2.4 Post Correspondence Problem

The Post Correspondence Problem (PCP), originally introduced by and shown to be undecidable by Post [4], has been used to show many problems arisen from formal languages are undecidable.

Formally, an instance of the PCP consists of a finite Σ , and a finite set $\{(u_i, v_i) \mid 1 \leq i \leq n\} \subseteq \Sigma^* \times \Sigma^*$ of n pairs of strings over Σ , deciding whether or not there exists word $j_1 j_2 \cdots j_k \in \{1, 2, \dots, n\}^+$ such that

$$u_{j_1} u_{j_2} \cdots u_{j_k} = v_{j_1} v_{j_2} \cdots v_{j_k}.$$

There are many variants of the PCP, for example, 2-Marked PCP [18] by Halava et al. However, the one of most convenience here is due to [16, 15], called “modified PCP”. Since the word $\omega \in \Sigma^*$ is of finite length², we assume that $m = \max\{|u_i|, |v_i|\}_{1 \leq i \leq n}$. We can put “o” into clearance between two letters of u_i (v_i), such that the resulting u'_i (v'_i) meets $|u'_i| = m$ ($|v'_i| = m$). Then the modified PCP problem is ask whether there exists $j_1 \cdots j_k \in \{1, \dots, n\}^+$ such that the equation $u'_{j_1} \cdots u'_{j_k} = v'_{j_1} \cdots v'_{j_k}$ holds after erasing all “o” in u'_i and v'_i .

3 Proof of Theorem 1.1

We are now proceeding to prove our main result.

Throughout this section, we fix $\Sigma = \{A, B, \circ\}$. We further fix the stack alphabet Γ of a constructed pBPA as follows

$$\Gamma = \{Z, Z', C, F, S, N, (x, y), X_{(x, y)}, G_i^j \mid (x, y) \in \Sigma \times \Sigma, 1 \leq i \leq n, 1 \leq j \leq m\}.$$

²We thank Dr. Forejt [19] for reminding us of that $|w| \in \mathbb{N}$ for any $w \in \Sigma^*$.

The elements in Γ also serve as symbols of atomic proposition whose senses will be clear later.

We construct the desirable stateless probabilistic pushdown process $\Delta = (\Gamma, \delta, \mathcal{P})$ in details.

Similar to [16, 15], our $pBPA$ Δ also works by two steps, the first of which is to guess a possible solution to a modified PCP instance by storing pairs of words (u_i, v_i) in the stack, which is achieved by the following transition rules (the probabilities of them are uniformly distributed):

$$\begin{aligned} Z &\rightarrow G_1^1 Z' \mid \cdots \mid G_n^1 Z'; \\ G_i^j &\rightarrow G_i^{j+1}(u_i(j), v_i(j)); \\ G_i^{m+1} &\rightarrow C \mid G_1^1 \mid \cdots \mid G_n^1. \end{aligned} \quad (1)$$

Obviously, we should let symbol Z serve as the initial stack symbol. When it begins to work, it firstly pushes $G_i^1 Z' \in \Gamma^*$ into stack with probability $\frac{1}{n}$. And then, the symbol in the top of the stack is G_i^1 (we read the stack from left to right). According to the above rules, G_i^1 is replaced by $G_i^2(u_i(1), v_i(1))$ with probability 1. The similar process will be continued until $G_i^{m+1}(u_i(m), v_i(m))$ are stored into the top of stack which means that the first pair of (u_i, v_i) is stored. After that, with probability $\frac{1}{n+1}$, Δ goes to push symbol C or G_i^1 into stack, depending on whether the procedure of guessing is at end or not. Of course, when the rule $G_i^{m+1} \rightarrow C$ is applied, it means Δ will go to check whether the pairs of words stored in the stack is a solution of a modified PCP instance. Obviously, the above guess procedure will lead to a word $j_1 j_2 \cdots j_k \in \{1, 2, \dots, n\}^+$ corresponding to the sequence of the words $(u_{j_1}, v_{j_1}), (u_{j_2}, v_{j_2}), \dots, (u_{j_k}, v_{j_k})$ pushed orderly into the stack. In addition, there is no other transition rules in ‘guessing-step’ for Δ except those illustrated by (1). From the above explanation, we readily see the following

Lemma 3.1 (Cf. [16], Lemma 3.2). *A configuration of the form $C\alpha$ is reachable from Z if and only if $\alpha \equiv (x_1, y_1) \cdots (x_l, y_l) Z'$ where $x_j, y_j \in \Sigma$, $1 \leq j \leq l$, and there is a word $j_1 j_2 \cdots j_k \in \{1, 2, \dots, n\}^+$ such that $x_1 \cdots x_l = u_{j_1} \cdots u_{j_k}$ and $y_l \cdots y_1 = v_{j_1} \cdots v_{j_k}$. \square*

The next step is for Δ to verify a stored pairs of words. Of course, to enable us to construct a PCTL formula describing this procedure, this step is slightly different from the one presented in [15, 16]. The transition rules (the probabilities of them are uniformly distributed) are as follows

$$\begin{aligned} C &\rightarrow N, & (x, y) &\rightarrow X_{(x,y)} \mid \epsilon, \\ N &\rightarrow F \mid S, & Z' &\rightarrow X_{(A,B)} \mid X_{(B,A)}, \\ F &\rightarrow \epsilon, & X_{(x,y)} &\rightarrow \epsilon, \\ S &\rightarrow \epsilon. \end{aligned} \quad (2)$$

Remark 3. *Once again, there is no other rules in ‘verifying-step’ for δ beside those described by (2). Compared to [16, 15], we have added an another symbol N into stack alphabet Γ , whose usage will be seen later on.*

When the stack symbol “ C ” is on the top of the stack, Δ is going to check whether the previous guess is a solution to the modified PCP instance. It first replaces C with N on the top of stack, with probability 1, and continue to push F or S into the stack, with probability $\frac{1}{2}$, depending on whether Δ wants to check u ’s or v ’s.

We employ the following two PCTL path formulas, which are from [15] (see, [15], p. 69)

$$\begin{aligned} \varphi_1 &\triangleq \left(\neg S \wedge \bigwedge_{z \in \Sigma} (\neg X_{(B,z)} \wedge \neg X_{(A,z)}) \right) \mathbf{U} \left(\bigvee_{z \in \Sigma} X_{(A,z)} \right) \\ \varphi_2 &\triangleq \left(\neg F \wedge \bigwedge_{z \in \Sigma} (\neg X_{(z,A)} \wedge \neg X_{(z,B)}) \right) \mathbf{U} \left(\bigvee_{z \in \Sigma} X_{(z,B)} \right). \end{aligned}$$

The following auxiliary Lemma is modified from (Lemma 4.4.8, [15], p. 45).

Lemma 3.2. Let ϑ and $\bar{\vartheta}$ be two functions from $\{A, B, Z'\}$ to $\{0, 1\}$, defined by

$$\begin{aligned}\vartheta(x) &= \bar{\vartheta}(x) = 1, & \text{if } x = Z'; \\ \vartheta(x) &= 1 - \bar{\vartheta}(x), & \text{if } x \in \{A, B\}.\end{aligned}\tag{3}$$

Let ρ and $\bar{\rho}$ be two functions from $\{A, B\}^+ Z'$ to $[0, 1]$ which are given by

$$\begin{aligned}\rho(x_1 x_2 \cdots x_n) &\triangleq \sum_{i=1}^n \vartheta(x_i) 2^{-i}; \\ \bar{\rho}(x_1 x_2 \cdots x_n) &\triangleq \sum_{i=1}^n \bar{\vartheta}(x_i) 2^{-i}.\end{aligned}$$

Then, for any $(u'_{j_1}, v'_{j_1}), (u'_{j_2}, v'_{j_2}), \dots, (u'_{j_k}, v'_{j_k}) \in \{A, B\}^+ \times \{A, B\}^+$,

$$u'_{j_1} u'_{j_2} \cdots u'_{j_k} = v'_{j_1} v'_{j_2} \cdots v'_{j_k}\tag{4}$$

if and only if

$$\rho(u'_{j_1} \cdots u'_{j_k} Z') + \bar{\rho}(v'_{j_1} \cdots v'_{j_k} Z') = 1\tag{5}$$

Proof. The “only if” part is obvious. Suppose that Eq. (4) holds and that $u'_{j_1} \cdots u'_{j_k} = y_1 \cdots y_l = v'_{j_1} \cdots v'_{j_k}$. Then we have

$$\begin{aligned}\rho(y_1 \cdots y_l Z') + \bar{\rho}(y_1 \cdots y_l Z') &= \sum_{i=1}^l (\vartheta(y_i) + \bar{\vartheta}(y_i)) \frac{1}{2^i} + (\vartheta(Z') + \bar{\vartheta}(Z')) \frac{1}{2^{l+1}} \\ &= \sum_{i=1}^l \frac{1}{2^i} + \frac{2}{2^{l+1}} = 1 \quad (\text{by (3)})\end{aligned}$$

The “if” part. If Eq. (5) fails, then

$$\rho(u'_{j_1} \cdots u'_{j_k} Z') + \bar{\rho}(v'_{j_1} \cdots v'_{j_k} Z') \neq 1$$

leads to that there is, at least, a y_h in $u'_{j_1} \cdots u'_{j_k}$ and a y'_h in $v'_{j_1} \cdots v'_{j_k}$ such that $\vartheta(y_h) + \bar{\vartheta}(y'_h) \neq 1$. By definition, $y_h \neq y'_h$. \square

By virtue of Lemma 3.2, we are ready to prove the following

Lemma 3.3. Let $\alpha = (u_{j_1}, v_{j_1})(u_{j_2}, v_{j_2}) \cdots (u_{j_k}, v_{j_k}) \in \Sigma^* \times \Sigma^*$ be the pair of words pushed into stack by Δ . Let (u'_i, v'_i) , $1 \leq i \leq j_k$, be the pair of words after erasing all “ \circ ” in u_i and v_i . Then

$$u'_{j_1} \cdots u'_{j_k} = v'_{j_1} \cdots v'_{j_k}\tag{6}$$

if and only if

$$\mathcal{M}_\Delta, N\alpha Z' \models^\nu \mathcal{P}_{=\frac{t}{2}}(\varphi_1) \wedge \mathcal{P}_{=\frac{1-t}{2}}(\varphi_2)$$

where $t : 0 < t < 1$ is a rational constant³.

³Here, t should not be considered as a free variable, and can not be 0 or 1. In fact, t is just the probability value of $\rho(u'_{j_1} \cdots u'_{j_k} Z')$. Furthermore, as there are many different α , there exist many different value of $\rho(u'_{j_1} \cdots u'_{j_k} Z')$ suitable for Lemma 3.3. We just use t denoting one among the possible values of $\rho(u'_{j_1} \cdots u'_{j_k} Z')$, leaving its specified numerical value unimportant. It is also very important that, t does not depend on the instance of modified PCP selected, see Lemma 3.2, the function ρ is irrelevant to the modified PCP instance.

Proof. It is clear that after α has been pushed in to the stack of Δ , the contents of stack is $C\alpha Z'$ (read from the left to right). Note that there is only one rule $C \rightarrow N$, which is applicable, thus, with probability 1, the content of stack changes in to $N\alpha Z'$. Obviously, there exist paths from N which goes through F , satisfying the PCTL path formula φ_1 and those from N which goes through S , satisfying the PCTL path formula φ_2 . The probabilities of paths from F satisfying φ_1 and of paths from S satisfying φ_2 are exactly $\rho(u'_{j_1} \cdots u'_{j_k} Z')$ and $\bar{\rho}(v'_{j_1} \cdots v'_{j_k} Z')$ respectively.

The “if” part. Suppose that

$$\mathcal{M}_\Delta, N\alpha Z' \models^\nu \mathcal{P}_{=\frac{t}{2}}(\varphi_1) \wedge \mathcal{P}_{=\frac{1-t}{2}}(\varphi_2).$$

Then the probability of paths from N , satisfying φ_1 is $\frac{t}{2}$, and the probability of paths from N , satisfying φ_2 is $\frac{1-t}{2}$. This leads to that the probability of paths from F , satisfying φ_1 is t , and the probability of paths from S , satisfying φ_2 is $1-t$, because $\mathcal{P}(N \rightarrow F) = \mathcal{P}(N \rightarrow S) = \frac{1}{2}$. Hence

$$\rho(u'_{j_1} \cdots u'_{j_k} Z') + \bar{\rho}(v'_{j_1} \cdots v'_{j_k} Z') = t + (1-t) = 1. \quad (7)$$

By Eq. (7) and Lemma 3.2, we conclude that Eq. (6) holds.

The “only if” part. Obviously, that Eq. (6) holds leads to

$$\rho(u'_{j_1} \cdots u'_{j_k} Z') + \bar{\rho}(v'_{j_1} \cdots v'_{j_k} Z') = 1 \Rightarrow \rho(u'_{j_1} \cdots u'_{j_k} Z') = 1 - \bar{\rho}(v'_{j_1} \cdots v'_{j_k} Z').$$

Namely, $\mathcal{P}(F\alpha Z' \models^\nu \varphi_1) = 1 - \mathcal{P}(S\alpha Z' \models^\nu \varphi_2)$, which further implies that

$$\mathcal{M}_\Delta, N\alpha Z' \models^\nu \mathcal{P}_{=\frac{t}{2}}(\varphi_1) \wedge \mathcal{P}_{=\frac{1-t}{2}}(\varphi_2).$$

The lemma follows. \square

The main result can now be proved as follows.

Proof of Theorem 1.1. Let ω be a path of $pBPA \Delta$, starting at C , induced by $C\alpha Z'$ —when α is guessed by Δ as a solution of the modified PCP instance.

Then, Lemma 3.3, together with the transition rule: $C \rightarrow N$, whose probability is 1, leads to the following

$$\begin{aligned} \text{Eq. (6) holds} &\Leftrightarrow \mathcal{M}_\Delta, \omega \models^\nu \mathbf{X} \left[\mathcal{P}_{=\frac{t}{2}}(\varphi_1) \wedge \mathcal{P}_{=\frac{1-t}{2}}(\varphi_2) \right] \quad (\text{by Lemma 3.3}) \\ &\Leftrightarrow \mathcal{M}_\Delta, C \models^\nu \mathcal{P}_{=1} \left(\mathbf{X} \left[\mathcal{P}_{=\frac{t}{2}}(\varphi_1) \wedge \mathcal{P}_{=\frac{1-t}{2}}(\varphi_2) \right] \right) \quad (\text{by } \mathcal{P}(C \rightarrow N) = 1) \\ &\Leftrightarrow \mathcal{M}_\Delta, Z \models^\nu \mathcal{P}_{>0} \left(\mathbf{true} \mathbf{U} \left(C \wedge \mathcal{P}_{=1} \left(\mathbf{X} \left[\mathcal{P}_{=\frac{t}{2}}(\varphi_1) \wedge \mathcal{P}_{=\frac{1-t}{2}}(\varphi_2) \right] \right) \right) \right) \end{aligned}$$

where the third “ \Leftrightarrow ” is by Lemma 3.1.

Thus

$$\mathcal{M}_\Delta, Z \models^\nu \mathcal{P}_{>0} \left(\mathbf{true} \mathbf{U} \left(C \wedge \mathcal{P}_{=1} \left(\mathbf{X} \left[\mathcal{P}_{=\frac{t}{2}}(\varphi_1) \wedge \mathcal{P}_{=\frac{1-t}{2}}(\varphi_2) \right] \right) \right) \right) \quad (8)$$

if and only if α is a solution of the modified PCP instance. Hence an algorithm for checking whether (8) is true, leads to an algorithm for the modified Post Correspondence Problem. \square

Remark 4. In fact, we can add a number of, but finite, N_i into the stack alphabet Γ , and a enough number of rules $C \rightarrow N_1 \rightarrow N_2 \rightarrow \cdots \rightarrow N_k \rightarrow N$ into δ . Hence, the following PCTL formula is also valid for the discussed problem

$$\mathcal{P}_{>0} \left(\mathbf{true} \mathbf{U} \left[C \wedge \mathcal{P}_{=1} \left(\mathbf{true} \mathbf{U} \mathcal{P}_{=1} \left[\mathbf{X} \left(\mathcal{P}_{=\frac{t}{2}}(\varphi_1) \wedge \mathcal{P}_{=\frac{1-t}{2}}(\varphi_2) \right) \right] \right) \right] \right)$$

Further, if we change the transition rule $C \rightarrow N$ to $C \rightarrow F \mid S$, the following formula is much simpler

$$\mathcal{P}_{>0} \left(\mathbf{true} \mathbf{U} \left[C \wedge \mathcal{P}_{=\frac{t}{2}}(\varphi_1) \wedge \mathcal{P}_{=\frac{1-t}{2}}(\varphi_2) \right] \right).$$

4 Conclusions

In the paper, it has shown that model-checking branching-time properties of stateless probabilistic pushdown process is undecidable, herein settling an open problem in [7]. However, there is another restricted version of probabilistic pushdown process which is more restricted than probabilistic pushdown process. In a word, it is a special probabilistic pushdown process but there is only one symbol in stack alphabet except the bottom symbol. This kind of process is the so-called *probabilistic one-counter process*, and its model-checking problem is still open, which deserves further consideration. Although the result in this paper indicates that there exists no exact algorithm for model-checking *pPBA* against PCTL, research method presented in [13] by Agrawal et al. may be applicable, which is an another interesting direction.

References

- [1] A.N. Shiryaev, Probability, (2nd Edition), Springer-Verlag, New York, 1995.
- [2] C. Baier, and J.P. Katoen, Principles of Model Checking, MIT Press, 2008.
- [3] E.M. Clarke, O. Grumberg, and D.A. Peled, Model Checking, MIT Press, 1999.
- [4] E.L. Post, A variant of a recursively unsolvable problem, Bulletin of the American Mathematical Society 52, 1946, pp. 264-268.
- [5] H. Hansson, and B. Jonsson, A logic for reasoning about time and reliability, Formal Aspects of Computing 6 (1994) 512-535.
- [6] J.E. Hopcroft, and J.D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, 1979.
- [7] J. Esparza, A. Kučera, and R. Mayr, Model-checking probabilistic pushdown automata, Logical Methods in Computer Science Vol. 2 (1:2) 2006, pp. 1-31.
- [8] J. Esparza, and A. Kučera, Model-checking probabilistic pushdown automata, Proceedings of LICS 2004, IEEE Computer Society Press, 2004, pp. 12-21.
- [9] J. Esparza, A. Kučera, and S. Schwoon, Model checking LTL with regular valuations for pushdown systems, Information and Computation 186, 2003, pp. 355-376.
- [10] M. Loève, Probability Theory I (4th edition), Springer-Verlag, New York, 1978.
- [11] M. Loève, Probability Theory II (4th edition), Springer-Verlag, New York, 1978.
- [12] M.Y. Vardi, Automatic verification of probabilistic concurrent finite-state programs, Proceedings of the 26th IEEE Annual Symposium on Foundations of Computer Science, 1985, pp. 327-338.
- [13] M. Agrawal, S. Akshay, B. Genest, and P.S. Thiagarajan. 2015. Approximate verification of the symbolic dynamics of Markov chains. J. ACM 62, 1, Article 2 (Feb. 2015), 34 pages.
- [14] S. Ginsburg, The Mathematical Theory of Context-Free Languages, McGraw-Hill, New York, 1966.
- [15] T. Brázdil, Verification of probabilistic recursive sequential programs, PhD thesis, Masaryk University, Faculty of Informatics, 2007.
- [16] T. Brázdil, V. Brožek, V. Forejt, and A. Kučera, Branching-time model-checking of probabilistic pushdown automata, Journal of Computer and System Sciences 80 (2014) 139-156.

- [17] T. Brázdil, A. Kčera, and O. Stražovský, On the decidability of temporal properties of probabilistic Pushdown automata, Proceedings of STACS 2005, Lecture Notes in Computer Science, vol. 3404, pp. 145-157.
- [18] V. Halava, M. Hirvensalo, and R. de Wolf, Decidability and Undecidability of Marked PCP, Proceedings of STACS 1999, Lecture Notes in Computer Science, vol. 1563, pp. 207-216.
- [19] V. Forejt, Private communication, Dec. 2013.